3D Reconstruction, Return of the Plushie

Miguel Mawyin
University of Toronto
1001137432
`miguel.mawyin@mail.utoronto.ca`

Shardul Bansal
University of Toronto
1004080040
`shardul.bansal@mail.utoronto.ca`

Conor Vedova
University of Toronto
1004405672
`conor.vedova@mail.utoronto.ca`

Abstract

The baked beans team focused on 3D Reconstruction. Our particular interest centers around Photo Tourism. We focused our experiments on constructing 3D models using techniques found from Photo Tourism papers [1]. In this work, we implemented 3D models for stereo images, for images with known relative camera positions, and finally for images with unknown camera positions but known camera parameters. Our work uses triangulation to estimate a point cloud for 2 images. This method is then followed by direct linear transformation (DLT), which estimates the relative position of new cameras. We achieved varying levels of results using these different methods, and we experimented with different procedures to see if we could improve them.

1 Introduction and Literature Review

1.1 Introduction to the problem

Our team is working on a problem within 3D Reconstruction, particularly Photo Tourism. We have focused our exploration of the field on a particular research paper within this subfield [1]. This focus has allowed us to not only narrow in on a problem, but it has provided us with a great starting point for 3D Reconstruction methods. Although we initially based our research around this paper, we needed to diverge from the original work for multiple reasons. First, we did not have a good dataset of instrinsic camera parameters which were required when using random photos from online. Second, the methods used in the paper had large amounts of runtime, sometimes taking two weeks to reconstruct one landmark [1]. So, we chose to diverge from this topic and instead work on alternative 3D Reconstruction methods which could help us focus in on the problem. Specifically, we focused on creating depth maps for stereo images, performing triangulation with known extrinsic parameters, performing extrinsic parameter estimation, and finally performing DLT to automate adding new images. We chose this approach as we wanted to work up from a simple problem to creating a complex solution.

1.2 Literature Review

We looked at a variety of different sources to better understand how to do 3D reconstruction well. We found the Photo Tourism paper interesting, trying to build 3D models of buildings using Structure from Motion (SfM) [1]. What peaked our interest was that SfM can produce point cloud based 3-D models similar to LiDAR. This technique can be used to create high resolution surface models. SfM also allows us to use consumer-grade digital cameras to obtain 3D Reconstructions. That's why we decided to incorporate it as part of our project to create 3D Reconstruction models. We knew, however, that the total running time for SfM will be a bottleneck, as the procedure can take over a day, mainly due to the iterative bundle adjustment talked about in the paper. The iterative bundle adjustment alongside the amount of coupling between cameras significantly increases the run-time of the algorithm.

To better help us recreate the results of the paper, we used this Medium article [2] that helps us understand SfM mathematically.

The paper from Snavely et. al discusses the following steps to obtain 3D models from a large dataset of images. The first step is to detect the feature points in each image. This is generally done through SIFT in most of the literature we read due to the good invariance that SIFT possesses to image transformations. The next step is to match feature points between different pairs of images. Then, the SfM procedure is used to recover a set of camera parameters. Since the datasets obtained do not carry information of the intrinsic parameters of the camera, this has to be estimated, and the SfM procedure allows us to estimate the relative position of each camera.

One of the most interesting papers we read was by Schonberger and Frahm. They revisted SfM, overcoming key challenges using the following methods that we found really intriguing:

- First, they introduced a geometric verification strategy that augments the scene graph with information. This greatly improves the performance of the triangulation components of SfM.
- Second, a next best view selection minimizing the reconstruction error of the 3D model.
- A more efficient bundle adjustment, improving the completeness of the reconstruction and minimizing granular irregularities [4].

We thought if we had ample time, we could extend the Photo-tourism SfM performance by implementing some of the aforementioned techniques.

2 Methodology

One of the first things we did was to generate a disparity map from 2 stereo images. The disparity map, alongside the camera calibration step, gives us the proper intrinsic parameters of the camera which can give us a depth map. From the depth map, we will also be able to generate a 3D mesh of the model in question. The disparity map we obtained from the stereo images is linked in the results section.

We realised that the primary challenge when working on 3D Reconstruction is to get the camera intrinsic and extrinsic parameters. This can be broken down into three matrices: **K** the intrinsic matrix, which holds the internal camera properties such as the focal length, principal point location, any scaling, etc; **R** the rotation matrix, which holds the rotation of the camera relative to the world coordinate; **T** the translation vector, which holds the location of the camera relative to the world coordinates.

Estimating three matrices at the same time can introducing a significant amount of error and it can be difficult to track which matrix estimation is doing poorly. We opted for a simpler incremental approach for estimating the matrices. First we found a dataset, by the University of Munich [5], containing images as well as the camera parameters. This allowed us to focus on triangulating matching keypoints.

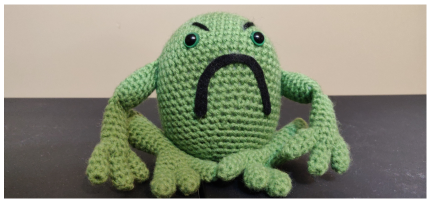
Once we were satisfied with our results from the triangulation algorithm we attempted to recover the extrinsic parameters of a camera which we knew the intrinsic matrix of. First we calibrated a camera using CV2.CALIBRATECAMERA and a calibration board to get the intrinsic matrix of the camera we were going to be using.

To get the extrinsic parameters we need the Essential Matrix E. Given two images, the Essential Matrix captures the relationship between correspondence points in the image place. Given two image correspondence points x and x', the Essential Matrix has the relation $x'^T Ex = 0$. We can estimate the Essential Matrix using CV2.FINDESSENTIALMATRIX, another possible way is using the Fundamental Matrix. The Fundamental Matrix is similar to the Essential Matrix, but captures the relationship using pixel coordinates. Once we have the Fundamental Matrix, we simply calculate $E = K'^T FK$, where K and K' are the intrinsic camera matrices.

Once the Essential Matrix has been estimated, it can be used to recover the relative position between the two cameras, as $E = [t]_\times R$, where $[t]_\times$ is the matrix representation of the dot product using the translation vector and R is the rotation between the two matrices. To recover the rotation and translation we can use an SVD decomposition of E, this method will yield two rotation matrices R_1, R_2 and a translation vector t, for a total of four possible relations: $(R_1, t), (R_1, -t), (R_2, t), (R_2, -t)$; however, only one of these pairs will triangulate points in front of the camera. Furthermore, the translation vector will not have the proper scale, as this can not be recovered using only the images. This is why CV2.RECOVERPOSE will return the translation vector with norm equal to one.

Another way to estimate the intrinsic and extrinsic parameters of the camera is using DLT

3 Results


(a) one of the stereo images

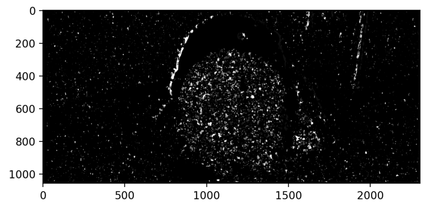

(b) disparity map generated

Figure 1: The stereo images and the corresponding disparity map

Look at the discussion of this result in the conclusion.

We had to first generate point clouds of the plushie. To do so, we would require a depth image where at each pixel location has a value proportional to the depth of the projected 3D scene point. Thus brighter points are further away from the camera. This was the method we had gotten from the class. However, since we didn't have the depth image, we generated the point clouds using 2 images. We demonstrated this here:

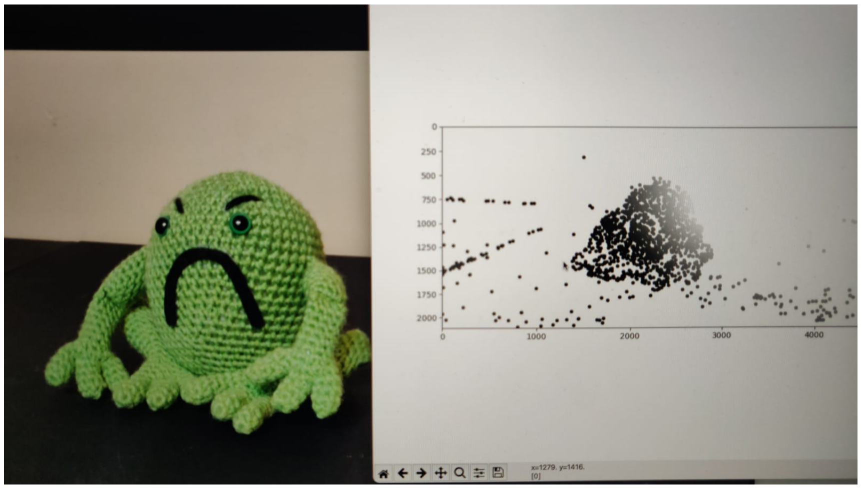


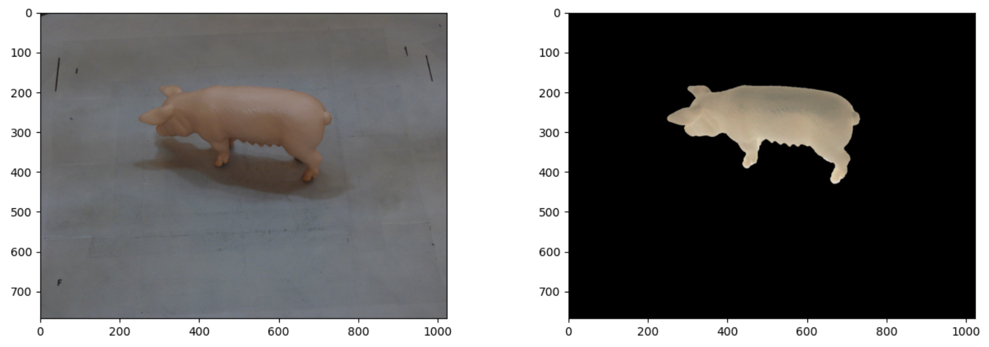
Figure 2: The plush toy on the left and the point cloud generated on the right

Furthermore, the plush toy image we are attempting to reconstruct does not have the best features for SIFT. For example, the toy is very round and does not have very sharp corners or recognizable feature points as other datasets might have. To rectify this, we may attempt to use a different set of images.

After realising that bundle adjustment will work far better using a dataset, we obtained a dataset of a pig and generated a point cloud using iterative bundle adjustment, giving us much better results than the pictures that we had taken ourselves.

3.1 Pig Triangulation

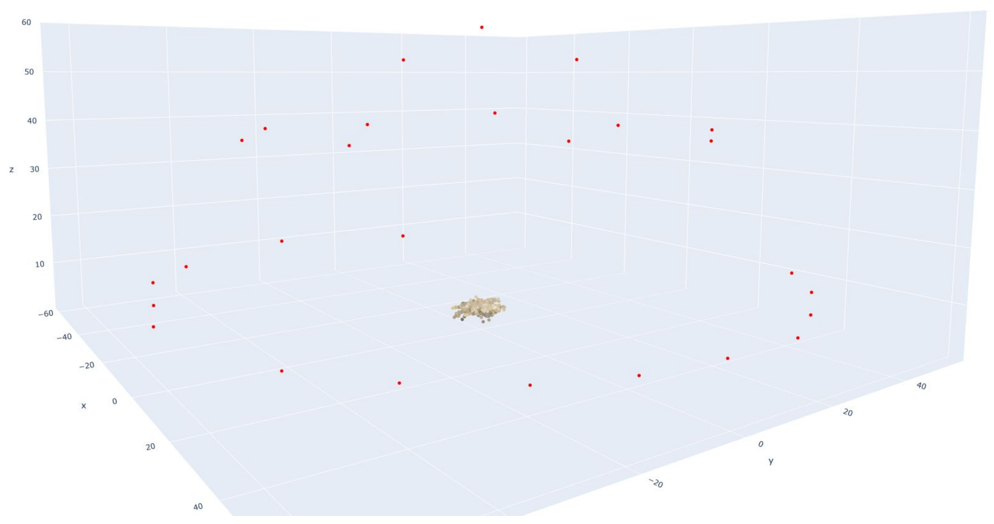
Our first attempt at 3D reconstruction was using the dataset from the University of Munich. The dataset contained the images of a model, the calibrated camera matrices, and silhouettes to remove background information in the images. Below are the images of the model, as well as the image after applying the silhouette:



(a) Pig model (b) Pig model after silhouette

Figure 3: Pig model images

From the silhouette images we use a SIFT procedure to find the keypoints and match them against every other image in the dataset. Since we had the camera parameters, triangulation was easy. We triangulate any keypoints which have at least 5 camera matches, this is to obtain a better result. Unfortunately in this process many of the keypoints do not get triangulated. Below is the result of the 3D reconstruction, the dots in red are the cameras around the model.



3.2 Extrinsic Parameters Estimation

Given the intrinsic matrix, we attempted to estimate the extrinsic parameters. First we compute the keypoints on the images, and then find the matches against the other images in the dataset. We estimate the Essential Matrix in this step, and remove any matching keypoints which do not follow the constraint defined by the Essential Matrix. From here we find the pair of images containing the most number of matches. Decomposing the Essential Matrix from this pair will give us the relative position between the cameras. One of the cameras will be placed at the origin. As mentioned above, the translation vector will have norm 1. We attempt to recover the proper scale of by the method described in the **Experiments** section, basically minimizing the reprojection error. Then we try to add another camera, we chose the camera which contains the most matches against the initial pair to add. Find the relative position of the new camera against the two cameras, and minimize the reprojection error to find the proper scale. We chose whichever position yielded the least amount of reprojection error. The algorithm continues in this iterative manner, adding images one by one. The result below was using 5 images and the house model:

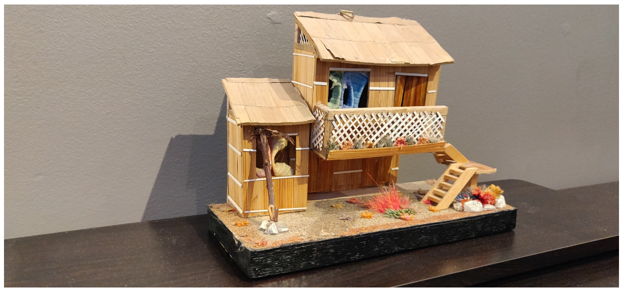


Figure 4: Original model of the house

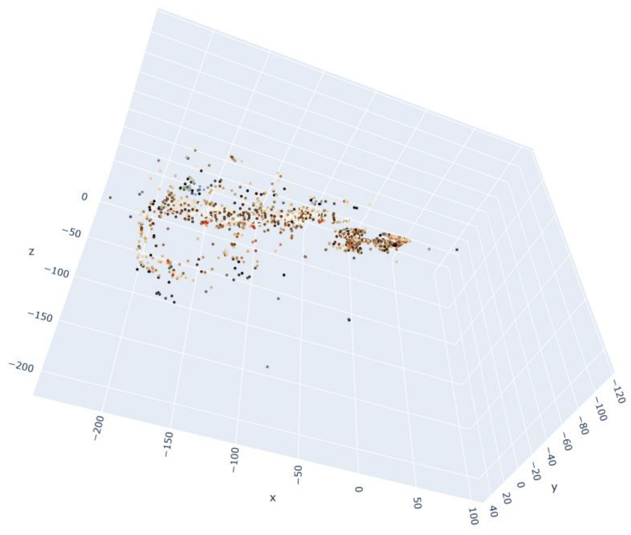
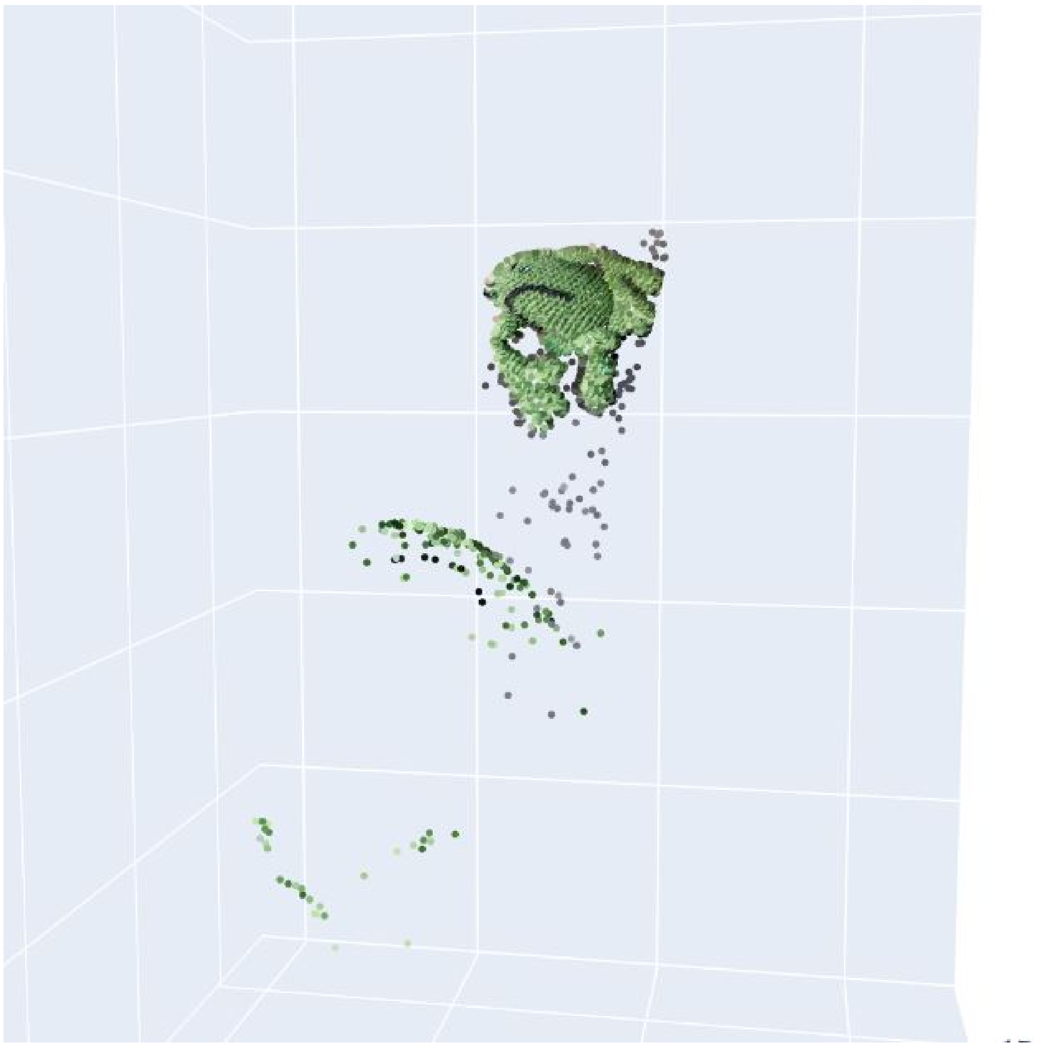


Figure 5: House with sharp edges and corners and its keypoints

3.3 Direct Linear Transformation and Triangulation

Our final attempt for 3D Reconstruction involved triangulation of the first two keyframes, or images. This method creates the initial point cloud that you can now add new image frames to. All new images are added with Direct Linear Transformation, a process by which you take known 3D points and the corresponding 2D points for the new image and calculate the matrix which transforms homogeneous 3D points to the new image's 2D points. You can then use the inverse of this matrix to add points to the point cloud. We applied this to the plushy toy and got the following pointcloud:



4 Experiments

4.1 Triangulation

We use the following method to triangulate points. Given a 3D point $Q = [X\ Y\ Z]^T$, let $\bar{Q}$ bet the augmented vector, i.e., $\bar{Q} = [X\ Y\ Z\ 1]^T$. Let C be the camera matrix, we have the following relation:

$$q = C\bar{Q}$$

$$\begin{bmatrix} wx \\ wy \\ w \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & c_{13} & c_{14} \\ c_{21} & c_{22} & c_{23} & c_{24} \\ c_{31} & c_{32} & c_{33} & c_{34} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} C1 & c_{14} \\ C2 & c_{24} \\ C3 & c_{34} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Where $C1, C2, C3$ are the compressed rows of the matrix C, without the last column. From this we get the following:

$$\begin{aligned} wx &= C1 \cdot Q + c_{14} \\ wy &= C2 \cdot Q + c_{24} \\ w &= C3 \cdot Q + c_{34} \end{aligned}$$

Plugging in w in the first two equations, and moving some terms we get the following system:

$$(xC3 - C1)Q = c_{14} - c_{34}x$$
$$(yC3 - C2)Q = c_{24} - c_{34}y$$

Which can be written as a familiar $AQ = b$, we can also keep adding rows to A and b in the case where there are multiple cameras matching the same point.

4.2 Extrinsic Parameter Estimation

To get the relative position of the cameras, CV2.RECOVERPOSE was used. As mentioned above, this returns the translation vector with unit length. To try an get a better estimate on the position of the image we minimize the reprojection error. We have the first camera projection matrix P1, the rotation R and translation T from decomposing the essential matrix, and a set of matching points from the images (pts1, pts2). We minimize the following function, using SCIPY.OPTIMIZE.MINIMIZE

Algorithm 1 Reprojection Error

```
Data: scale
Result: scale that gives the least projection error
  P2 = camera_compose(K2, R, scale * T)
  3D_Points = triangulate(pts1, pts2, P1, P2)
  return projection_error(pts1, 3D_points) + projection_error(pts2, 3D_points)
```

4.3 Direct Linear Transformation

As our Triangulation methods were not working out perfectly, we decided to use direct linear transformation to add new images besides the initial two. The process goes as follows:

1. Use Triangulation to create an initial point cloud for the first two images 2. Match points in the new image to points already in the point cloud (2D points must of course be matched, not 3D). 3. If there are enough points to solve DLT (6 points are required to have a unique solution), then apply DLT to get a calibration matrix. Then, use the inverse of this matrix to add some 2D points from the new image to the point cloud. 4. If there are not enough point matches, then create a new keyframe and apply triangulation.

Step 3 requires you to perform SVD on the following matrix, where (X,Y,Z) is a coordinate in the 3D world coordinate system and (x,y) is a coordinate in the 2D image coordinate system:

$$A = \begin{bmatrix} -X & -Y & -Z & -1 & 0 & 0 & 0 & 0 & x*X & x*Y & x*Z & x \\ 0 & 0 & 0 & 0 & -X & -Y & -Z & -1 & y*X & y*Y & y*Z & y \end{bmatrix}$$

The SVD of matrix A = USV. The final column of matrix V will be the parameters that you need. Transform this column into a 3x4 matrix and this matrix will map 4D world homogeneous coordinates to 3D image homogeneous coordinates. In order to obtain the 2D image coordinates, simply divide the vector by the final entry of the vector and drop the last entry.

5 Conclusion

A large majority of the avenues that we pursued did not yield fruitful results. We feel this is largely due to the open-ended nature of the 3D reconstruction problem. At the same time, because of the open-ended nature of it, we were able to try a lot of different approaches. We will use this space to discuss the successes and failures of stereo $\rightarrow$ disparity images, triangulation and extrinsic parameter estimation and lastly direct linear transformations.

There's clearly issues with the disparity map in Figure 1. There's far too much noise and the key shapes are not being able to be detected well. After fiddling with the parameters of the number of disparities as well as the block size, we did not obtain much better results. Nonetheless, we were required to calculate the intrinsic and extrinsic parameters to be able to generate a depth map, which can only ever be as good as the disparity map. Since the disparity map yielded poor results, we chose to follow a different route. One of the other reasons we felt that SIFT performed rather poorly was due to the lack of corners and edges in the actual subject of the image. There were scarce key-points that could be used to estimate a 3D model. However, this improved as we used other subjects in our experimentation.

When estimating the translation scale using a minimization procedure can lead to more error. If the optimizer goes on a bad path it can lead to return of a scale which is magnitudes higher than what is expected. So bounds must be placed on the optimizer. Futhermore, using this method for multiple images can increase the error more and more as images get added. A solution to this problem is to use DLT for adding multiple images to the point cloud. However, this does not fix the issue that reconstruction error increases as more images are added. A possible route in the future is to explore methods to reduce reconstruction error such as Bundle Adjustment and to improve the efficiency of our pipelines.

6 Author's Contributions

Conor Vedova created the pipeline for Direct Linear Transformation. He also experimented with different methods such as depth maps for stereo images.

Shardul Bansal created the disparity images using stereo images and experimented with point cloud mappings.

Miguel Mawyin created the triangulation method and worked on extrinsic parameter estimation.

References

[1] Noah Snavely, Steven M. Seitz, Richard Szeliski, "Photo tourism: Exploring photo collections in 3D," ACM Transactions on Graphics (SIGGRAPH Proceedings), 25(3), 2006, 835-846.

[2] MADALI, Nabil. "Structure from Motion." Medium, Towards Data Science, 30 July 2020, towardsdatascience.com/structure-from-motion-311c0cb50e8d.

[3] Snavely, Noah, Steven M. Seitz, and Richard Szeliski. "Modeling the world from internet photo collections." International journal of computer vision 80.2 (2008): 189-210.

[4] Johannes L. Schonberger, Jan-Michael Frahm; Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 4104-4113

[5] Computer Vision Group, TUM Department of Informatics, Technical University of Munich. (2020, November 18). Multiview Datasets. https://vision.in.tum.de/data/datasets/3dreconstruction